

In today's AI-driven B2B SaaS landscape, especially with companies like **PayPal** integrating AI for fraud detection, **H2O.ai** pioneering open-source AI platforms, and frameworks like **PM Toolkit** equipping product managers with decision-calculators, the rush to pick the "best" model can quickly devolve into a never-ending showdown of metrics and marketing. But as someone with 12 years in B2B SaaS product management — including 4 years shipping AI-assisted workflows — I've learned to see through the noise. Model comparisons aren't battles to claim dominance but rather informed choices aligned with real user workflows and business goals.

The Model Is Not the Product

It may sound obvious, but it's a fundamental point too often overlooked. A *model* is just one cog in a much broader product system. If you think of an AI feature like a module delivering insights, the product is the entire experience: user interface, reliability, workflows, monitoring, and yes, the fallback and kill switch plan when AI falters.

What do users do today? Before diving into model comparisons, map out the current workflow your users follow. This grounds your evaluation in reality and cuts through hype. For example, if you are at **PayPal** building AI to flag suspicious transactions, understanding how manual review currently operates is key. Will the AI augment or automate steps? What are the failure modes most impactful to users?

Why This Matters

- It stops you from chasing incremental accuracy claims that don't improve the actual user experience.
- Sets expectations around intervention points, kill switches, and error recovery.
- Helps prioritize model qualities beyond benchmark accuracy — such as latency, explainability, and maintainability.

Define an Eval Before Release

One of the most powerful habits I've developed over the years is writing evaluations as user stories — not lab prompt snippets. Before training or selecting any model, define the **eval on your task** clearly. This means:

1. Identifying specific user scenarios you want the model to support.
2. Listing expected inputs, outputs, and failure modes.
3. Determining measurable success criteria relevant to your product's metrics (think: precision for fraud alerts, latency for chatbots, etc.).

Once these evals are laid out, implement them as lightweight checks — often in spreadsheets for quick iteration — that can be reused across models and versions. The **PM Toolkit** emphasizes establishing these baselines early to help with decision-making on whether to build, buy, or switch models.



Examples of Eval Metrics & Tools

- **Business impact calculations:** Use *PM Toolkit* calculators for TAM/SAM/SOM, ROI, LTV, CAC, retention, and churn metrics.
- **Model performance:** Evaluate metrics aligned to your product goals: precision/recall, false positive rates, latency, robustness to noisy inputs.
- **User feedback & NPS:** Combine quantitative data with qualitative insights via NPS and PMF score calculators.
- **Experiment design:** Leverage sample size calculators and A/B test tools from *PM Toolkit* before wide rollout.

Task-dependent Model Behavior and Failure Modes

Before falling into a “model wars” trap where two models are compared solely on accuracy or benchmark performance, remember that different models showcase a distinct pattern of behaviors depending on your task.

For instance, a model that excels at generating fluent text (e.g., GPT-based) may struggle with domain-specific data extraction compared to parameter-efficient fine-tuned models like some H2O.ai offerings. Conversely, specialized models might have lower coverage or more brittle outputs in edge cases.

Effective comparison means:

- **Testing models on your specific task:** Run real user queries or inputs and review outputs critically.
- **Cataloging failure modes:** Document where each model fails — is it hallucinating? Missing critical data? Producing biased outputs?
- **Evaluating automation & augmentation scenarios:** Which tasks can fully automate without human oversight, and where should the model assist, leaving the final call to users?

With **PayPal**, for example, automation vs augmentation may mean fully auto-blocking likely fraud or simply flagging as higher priority cases for review. The right model choice depends on these workflows and their risk tolerance.

Automation vs Augmentation vs Innovation Patterns

Understanding the pattern that your AI feature fits into is crucial. The three common AI adoption patterns are:

1. **Automation:** Replacing a repetitive manual step end-to-end. Requires high confidence and robust kill switches.
2. **Augmentation:** Enhancing human judgment with suggestions or prioritized hints. Favors models with conservative failure modes.
3. **Innovation:** Creating entirely new workflows or products made possible by AI capabilities.

Each of these patterns demands different model evaluation approaches. For pure automation, fail-safe thresholds and fallback modes are non-negotiable. For augmentation, user trust and transparent uncertainty are paramount. Innovation allows more experimentation with model types but still needs user-centered evals.

H2O.ai, for pmtoolkit.ai example, provides AutoML tools that product teams can use to iterate rapidly through different models and workflows — helping them identify which pattern fits best before locking in choices.

How to Avoid Getting Stuck in Model Wars

The so-called “model wars” — endless debates over which architecture or training trick is superior — are a red herring when detached from workflow context and business goals. Here’s what I do to avoid that trap:

1. **Start with the user and workflow:** What problem are you solving? How do users do it today?
2. **Define a baseline and a clear eval suite:** You can’t measure improvement without a stable, task-aligned benchmark.
3. **Set explicit ship/no-ship thresholds:** I keep a personal checklist that combines technical metrics with business impact and operational risk to decide readiness.
4. **Use tools to quantify impact:** Incorporate *PM Toolkit calculators* for TAM, ROI, RICE prioritization, retention and churn impact, and experiment design before full rollout.
5. **Experiment with switch-later in mind:** Don’t build brittle dependencies on a specific model or vendor. Design your stack to allow swapping models without product overhaul.

6. **Monitor in production:** Use real-time dashboards and error budgets. Include kill switches that allow rolling back models swiftly if failure modes appear.

Using Spreadsheets for Practical Evals

You don't need a fancy ML ops platform to start. I've shipped products leveraging simple spreadsheets as a lightweight, reusable eval framework. Here's how:

- Collect user input-output pairs and manually label success/failure as you update models.
- Track numeric metrics alongside qualitative notes on early failure modes.
- Use built-in calculators from PM Toolkit templates for size, impact, and sample size calculations.
- Visualize trends over time to detect regression and improvements in a transparent, shareable way.

This approach fosters a collaborative evaluation process between product, data science, and business stakeholders — minimizing subjective "model wars" narrative and centering on what truly matters.

Key Takeaways

Principle Description The Model Is Not the Product Focus on user workflows and product experience first, then select models that fit seamlessly. Eval on Your Task Develop clear, task-specific evals prior to model training or selection. Catalog Task-dependent Failure Modes Know where each model breaks on your inputs to make informed tradeoffs. Automation vs Augmentation vs Innovation Match your AI pattern to the right model evaluation focus and risk strategy. Set Ship/No-Ship Thresholds and Use PM Toolkit Anchor decisions in business impact, risk, and experiment design using PM Toolkit calculators. Design for Switch Later Keep your product architecture flexible to swap models without disruption.

Final Thoughts

Companies like **PayPal** and **H2O.ai** demonstrate that the magic isn't just in the underlying models but in thoughtfully integrating them into workflows that users trust and find valuable. Armed with clear tasks, measurable evals, and pragmatic decision tools like those found in the **PM Toolkit**, product teams can steer clear of pointless model wars and instead deliver AI solutions that genuinely work and scale.

Remember: pick a model fit for your task, baseline comparisons with your real data, and always design with the possibility to **switch later**. The AI race isn't won by individual models but by flexible, user-first products evolving iteratively.